autogluon_each_location

October 7, 2023

[1]:
```
import pandas as pd
import numpy as np



import warnings
warnings.filterwarnings("ignore")

def fix_datetime(X, name):
    # Convert 'date_forecast' to datetime format and replace original column␣
 ↪with 'ds'
    X['ds'] = pd.to_datetime(X['date_forecast'])
    X.drop(columns=['date_forecast'], inplace=True, errors='ignore')
    X.sort_values(by='ds', inplace=True)
    X.set_index('ds', inplace=True)

    # Drop rows where the minute part of the time is not 0
    X = X[X.index.minute == 0]
    return X



def convert_to_datetime(X_train_observed, X_train_estimated, X_test, y_train):
    X_train_observed = fix_datetime(X_train_observed, "X_train_observed")
    X_train_estimated = fix_datetime(X_train_estimated, "X_train_estimated")
    X_test = fix_datetime(X_test, "X_test")


    X_train_observed["estimated_diff_hours"] = 0
    X_train_estimated["estimated_diff_hours"] = (X_train_estimated.index - pd.
 ↪to_datetime(X_train_estimated["date_calc"])).dt.total_seconds() / 3600
    X_test["estimated_diff_hours"] = (X_test.index - pd.
 ↪to_datetime(X_test["date_calc"])).dt.total_seconds() / 3600

    X_train_estimated["estimated_diff_hours"] =␣
 ↪X_train_estimated["estimated_diff_hours"].astype('int64')
    # the filled once will get dropped later anyways, when we drop y nans
```

```
    X_test["estimated_diff_hours"] = X_test["estimated_diff_hours"].fillna(-50).
 ↪astype('int64')


    X_train_estimated.drop(columns=['date_calc'], inplace=True)
    X_test.drop(columns=['date_calc'], inplace=True)

    y_train['ds'] = pd.to_datetime(y_train['time'])
    y_train.drop(columns=['time'], inplace=True)
    y_train.sort_values(by='ds', inplace=True)
    y_train.set_index('ds', inplace=True)

    return X_train_observed, X_train_estimated, X_test, y_train




def preprocess_data(X_train_observed, X_train_estimated, X_test, y_train,␣
 ↪location):
    # convert to datetime
    X_train_observed, X_train_estimated, X_test, y_train =␣
 ↪convert_to_datetime(X_train_observed, X_train_estimated, X_test, y_train)

    y_train["y"] = y_train["pv_measurement"].astype('float64')
    y_train.drop(columns=['pv_measurement'], inplace=True)
    X_train = pd.concat([X_train_observed, X_train_estimated])


    # clip all y values to 0 if negative
    y_train["y"] = y_train["y"].clip(lower=0)

    X_train = pd.merge(X_train, y_train, how="outer", left_index=True,␣
 ↪right_index=True)

    X_train["location"] = location
    X_test["location"] = location

    return X_train, X_test
# Define locations
locations = ['A', 'B', 'C']

X_trains = []
X_tests = []
# Loop through locations
for loc in locations:
    print(f"Processing location {loc}...")
    # Read target training data
```

```
    y_train = pd.read_parquet(f'{loc}/train_targets.parquet')

    # Read estimated training data and add location feature
    X_train_estimated = pd.read_parquet(f'{loc}/X_train_estimated.parquet')

    # Read observed training data and add location feature
    X_train_observed= pd.read_parquet(f'{loc}/X_train_observed.parquet')

    # Read estimated test data and add location feature
    X_test_estimated = pd.read_parquet(f'{loc}/X_test_estimated.parquet')

    # Preprocess data
    X_train, X_test = preprocess_data(X_train_observed, X_train_estimated,␣
 ↪X_test_estimated, y_train, loc)

    X_trains.append(X_train)
    X_tests.append(X_test)

# Concatenate all data and save to csv
X_train = pd.concat(X_trains)
X_test = pd.concat(X_tests)
```

```
Processing location A…
Processing location B…
Processing location C…
```

1 Feature enginering

[4]:
```
# temporary
X_train["hour"] = X_train.index.hour
X_train["weekday"] = X_train.index.weekday
# weekday or is_weekend
X_train["is_weekend"] = X_train["weekday"].apply(lambda x: 1 if x >= 5 else 0)

# drop weekday
#X_train.drop(columns=["weekday"], inplace=True)
X_train["month"] = X_train.index.month
X_train["year"] = X_train.index.year

X_test["hour"] = X_test.index.hour
X_test["weekday"] = X_test.index.weekday

# weekday or is_weekend
X_test["is_weekend"] = X_test["weekday"].apply(lambda x: 1 if x >= 5 else 0)

# drop weekday
#X_test.drop(columns=["weekday"], inplace=True)
```

```
X_test["month"] = X_test.index.month
X_test["year"] = X_test.index.year



to_drop = ["snow_drift:idx", "snow_density:kgm3"]

X_train.drop(columns=to_drop, inplace=True)
X_test.drop(columns=to_drop, inplace=True)

X_train.dropna(subset=['y'], inplace=True)
X_train.to_csv('X_train_raw.csv', index=True)
X_test.to_csv('X_test_raw.csv', index=True)
```

[5]:
```
import autogluon.eda.auto as auto
auto.dataset_overview(train_data=X_train, test_data=X_test, label="y",␣
↪sample=None)
```

train_data dataset summary

	count	unique	top	freq	mean	\
absolute_humidity_2m:gm3	92951	165			6.017608	
air_density_2m:kgm3	92951	293			1.255435	
ceiling_height_agl:m	72276	40993			2802.588135	
clear_sky_energy_1h:J	92951	48602			515154.09375	
clear_sky_rad:W	92951	7815			143.101379	
cloud_base_agl:m	84404	34862			1692.934692	
dew_or_rime:idx	92951	3			0.007025	
dew_point_2m:K	92951	436			275.237762	
diffuse_rad:W	92951	2870			39.495815	
diffuse_rad_1h:J	92951	48553			142180.03125	
direct_rad:W	92951	5296			50.205021	
direct_rad_1h:J	92951	41885			180740.1875	
effective_cloud_cover:p	92951	1001			67.013519	
elevation:m	92951	3			11.401738	
estimated_diff_hours	92951	26			3.143516	
fresh_snow_12h:cm	92951	125			0.116175	
fresh_snow_1h:cm	92951	39			0.00963	
fresh_snow_24h:cm	92951	161			0.229894	
fresh_snow_3h:cm	92951	70			0.029001	
fresh_snow_6h:cm	92951	96			0.058069	
hour	93024	24			11.501462	
is_day:idx	92951	2			0.483341	
is_in_shadow:idx	92951	2			0.565384	
location	93024	3	A	34085		
month	93024	12			6.290484	
msl_pressure:hPa	92951	874			1009.502563	
precip_5min:mm	92951	64			0.005674	

precip_type_5min:idx	92951	7	0.083259
pressure_100m:hPa	92951	888	995.81897
pressure_50m:hPa	92951	897	1001.949646
prob_rime:p	92951	700	0.756834
rain_water:kgm2	92951	11	0.009677
relative_humidity_1000hPa:p	92951	788	73.669556
sfc_pressure:hPa	92951	902	1008.107849
snow_depth:cm	92951	165	0.193203
snow_melt_10min:mm	92951	19	0.000275
snow_water:kgm2	92951	42	0.090324
sun_azimuth:d	92951	69692	182.386337
sun_elevation:d	92951	49376	-1.207574
super_cooled_liquid_water:kgm2	92951	15	0.056944
t_1000hPa:K	92951	447	279.431061
total_cloud_cover:p	92951	1001	73.604263
visibility:m	92951	85686	33027.933594
weekend	93024	2	0.28655
wind_speed_10m:ms	92951	119	3.037911
wind_speed_u_10m:ms	92951	188	0.662565
wind_speed_v_10m:ms	92951	167	0.6824
wind_speed_w_1000hPa:ms	92951	3	-0.000016
y	93024	12430	287.019652
year	93024	6	2020.69495

	std	min	25%	\
absolute_humidity_2m:gm3	2.714546	0.5	4.0	
air_density_2m:kgm3	0.036608	1.139	1.23	
ceiling_height_agl:m	2521.408447	27.799999	1037.099976	
clear_sky_energy_1h:J	820525.5	0.0	0.0	
clear_sky_rad:W	228.507324	0.0	0.0	
cloud_base_agl:m	1790.963745	27.4	572.200012	
dew_or_rime:idx	0.246032	-1.0	0.0	
dew_point_2m:K	6.83461	247.300003	270.700012	
diffuse_rad:W	60.647518	0.0	0.0	
diffuse_rad_1h:J	215907.21875	0.0	0.0	
direct_rad:W	112.946068	0.0	0.0	
direct_rad_1h:J	401735.03125	0.0	0.0	
effective_cloud_cover:p	35.044811	0.0	41.299999	
elevation:m	7.877236	6.0	6.0	
estimated_diff_hours	8.935328	0.0	0.0	
fresh_snow_12h:cm	0.780374	0.0	0.0	
fresh_snow_1h:cm	0.112621	0.0	0.0	
fresh_snow_24h:cm	1.218249	0.0	0.0	
fresh_snow_3h:cm	0.28067	0.0	0.0	
fresh_snow_6h:cm	0.481389	0.0	0.0	
hour	6.92022	0.0	6.0	
is_day:idx	0.499725	0.0	0.0	
is_in_shadow:idx	0.495709	0.0	0.0	

location			
month	3.587269	1.0	3.0
msl_pressure:hPa	13.089046	944.299988	1001.400024
precip_5min:mm	0.033511	0.0	0.0
precip_type_5min:idx	0.384904	0.0	0.0
pressure_100m:hPa	13.008334	929.799988	987.799988
pressure_50m:hPa	13.067102	935.599976	993.900024
prob_rime:p	5.434649	0.0	0.0
rain_water:kgm2	0.042968	0.0	0.0
relative_humidity_1000hPa:p	14.328553	19.5	64.199997
sfc_pressure:hPa	13.128181	941.400024	1000.0
snow_depth:cm	1.254293	0.0	0.0
snow_melt_10min:mm	0.004312	-0.0	-0.0
snow_water:kgm2	0.250991	0.0	0.0
sun_azimuth:d	102.913605	0.008	92.794006
sun_elevation:d	24.010485	-49.979	-18.511
super_cooled_liquid_water:kgm2	0.111482	0.0	0.0
t_1000hPa:K	6.520342	257.899994	274.899994
total_cloud_cover:p	34.993042	0.0	51.700001
visibility:m	18319.150391	130.600006	15798.950195
weekend	0.452152	0.0	0.0
wind_speed_10m:ms	1.778505	0.0	1.7
wind_speed_u_10m:ms	2.808995	-7.3	-1.4
wind_speed_v_10m:ms	1.896996	-9.3	-0.6
wind_speed_w_1000hPa:ms	0.006502	-0.1	0.0
y	766.407785	-0.0	0.0
year	1.187172	2018.0	2020.0

	50%	75%	max	\
absolute_humidity_2m:gm3	5.4	7.8	17.5	
air_density_2m:kgm3	1.255	1.279	1.441	
ceiling_height_agl:m	1803.25	3814.824951	12431.299805	
clear_sky_energy_1h:J	4544.899902	778247.25	3006697.25	
clear_sky_rad:W	0.0	220.949997	835.299988	
cloud_base_agl:m	1128.550049	2016.699951	11688.900391	
dew_or_rime:idx	0.0	0.0	1.0	
dew_point_2m:K	275.0	280.5	293.799988	
diffuse_rad:W	0.0	66.0	340.100006	
diffuse_rad_1h:J	9951.700195	236502.75	1182265.375	
direct_rad:W	0.0	29.0	684.299988	
direct_rad_1h:J	0.0	113366.25	2445897.0	
effective_cloud_cover:p	80.800003	99.300003	100.0	
elevation:m	7.0	24.0	24.0	
estimated_diff_hours	0.0	0.0	39.0	
fresh_snow_12h:cm	0.0	0.0	37.400002	
fresh_snow_1h:cm	0.0	0.0	7.1	
fresh_snow_24h:cm	0.0	0.0	37.400002	
fresh_snow_3h:cm	0.0	0.0	20.6	

fresh_snow_6h:cm	0.0	0.0	34.0
hour	12.0	17.0	23.0
is_day:idx	0.0	1.0	1.0
is_in_shadow:idx	1.0	1.0	1.0
location			
month	6.0	10.0	12.0
msl_pressure:hPa	1010.299988	1018.599976	1044.099976
precip_5min:mm	0.0	0.0	1.38
precip_type_5min:idx	0.0	0.0	6.0
pressure_100m:hPa	996.799988	1004.900024	1030.900024
pressure_50m:hPa	1002.900024	1011.099976	1037.300049
prob_rime:p	0.0	0.0	97.199997
rain_water:kgm2	0.0	0.0	1.4
relative_humidity_1000hPa:p	76.0	85.099998	100.0
sfc_pressure:hPa	1009.0	1017.200012	1043.800049
snow_depth:cm	0.0	0.0	18.299999
snow_melt_10min:mm	0.0	-0.0	0.18
snow_water:kgm2	0.0	0.1	6.9
sun_azimuth:d	179.526001	271.503479	359.997009
sun_elevation:d	-0.99	15.538	49.917999
super_cooled_liquid_water:kgm2	0.0	0.1	1.4
t_1000hPa:K	278.700012	283.899994	303.299988
total_cloud_cover:p	94.800003	100.0	100.0
visibility:m	37350.300781	48679.550781	76737.796875
weekend	0.0	1.0	1.0
wind_speed_10m:ms	2.7	4.1	15.2
wind_speed_u_10m:ms	0.3	2.5	12.2
wind_speed_v_10m:ms	0.7	1.9	9.0
wind_speed_w_1000hPa:ms	0.0	0.0	0.1
y	0.0	172.92	5733.42
year	2021.0	2022.0	2023.0

	dtypes	missing_count	missing_ratio	raw_type	\
absolute_humidity_2m:gm3	float32	73	0.000785	float	
air_density_2m:kgm3	float32	73	0.000785	float	
ceiling_height_agl:m	float32	20748	0.223039	float	
clear_sky_energy_1h:J	float32	73	0.000785	float	
clear_sky_rad:W	float32	73	0.000785	float	
cloud_base_agl:m	float32	8620	0.092664	float	
dew_or_rime:idx	float32	73	0.000785	float	
dew_point_2m:K	float32	73	0.000785	float	
diffuse_rad:W	float32	73	0.000785	float	
diffuse_rad_1h:J	float32	73	0.000785	float	
direct_rad:W	float32	73	0.000785	float	
direct_rad_1h:J	float32	73	0.000785	float	
effective_cloud_cover:p	float32	73	0.000785	float	
elevation:m	float32	73	0.000785	float	
estimated_diff_hours	float64	73	0.000785	float	

fresh_snow_12h:cm	float32	73	0.000785	float
fresh_snow_1h:cm	float32	73	0.000785	float
fresh_snow_24h:cm	float32	73	0.000785	float
fresh_snow_3h:cm	float32	73	0.000785	float
fresh_snow_6h:cm	float32	73	0.000785	float
hour	int64			int
is_day:idx	float32	73	0.000785	float
is_in_shadow:idx	float32	73	0.000785	float
location	object			object
month	int64			int
msl_pressure:hPa	float32	73	0.000785	float
precip_5min:mm	float32	73	0.000785	float
precip_type_5min:idx	float32	73	0.000785	float
pressure_100m:hPa	float32	73	0.000785	float
pressure_50m:hPa	float32	73	0.000785	float
prob_rime:p	float32	73	0.000785	float
rain_water:kgm2	float32	73	0.000785	float
relative_humidity_1000hPa:p	float32	73	0.000785	float
sfc_pressure:hPa	float32	73	0.000785	float
snow_depth:cm	float32	73	0.000785	float
snow_melt_10min:mm	float32	73	0.000785	float
snow_water:kgm2	float32	73	0.000785	float
sun_azimuth:d	float32	73	0.000785	float
sun_elevation:d	float32	73	0.000785	float
super_cooled_liquid_water:kgm2	float32	73	0.000785	float
t_1000hPa:K	float32	73	0.000785	float
total_cloud_cover:p	float32	73	0.000785	float
visibility:m	float32	73	0.000785	float
weekend	int64			int
wind_speed_10m:ms	float32	73	0.000785	float
wind_speed_u_10m:ms	float32	73	0.000785	float
wind_speed_v_10m:ms	float32	73	0.000785	float
wind_speed_w_1000hPa:ms	float32	73	0.000785	float
y	float64			float
year	int64			int

	variable_type	special_types
absolute_humidity_2m:gm3	numeric	
air_density_2m:kgm3	numeric	
ceiling_height_agl:m	numeric	
clear_sky_energy_1h:J	numeric	
clear_sky_rad:W	numeric	
cloud_base_agl:m	numeric	
dew_or_rime:idx	category	
dew_point_2m:K	numeric	
diffuse_rad:W	numeric	
diffuse_rad_1h:J	numeric	
direct_rad:W	numeric	

direct_rad_1h:J	numeric
effective_cloud_cover:p	numeric
elevation:m	category
estimated_diff_hours	numeric
fresh_snow_12h:cm	numeric
fresh_snow_1h:cm	numeric
fresh_snow_24h:cm	numeric
fresh_snow_3h:cm	numeric
fresh_snow_6h:cm	numeric
hour	numeric
is_day:idx	category
is_in_shadow:idx	category
location	category
month	category
msl_pressure:hPa	numeric
precip_5min:mm	numeric
precip_type_5min:idx	category
pressure_100m:hPa	numeric
pressure_50m:hPa	numeric
prob_rime:p	numeric
rain_water:kgm2	category
relative_humidity_1000hPa:p	numeric
sfc_pressure:hPa	numeric
snow_depth:cm	numeric
snow_melt_10min:mm	category
snow_water:kgm2	numeric
sun_azimuth:d	numeric
sun_elevation:d	numeric
super_cooled_liquid_water:kgm2	category
t_1000hPa:K	numeric
total_cloud_cover:p	numeric
visibility:m	numeric
weekend	category
wind_speed_10m:ms	numeric
wind_speed_u_10m:ms	numeric
wind_speed_v_10m:ms	numeric
wind_speed_w_1000hPa:ms	category
y	numeric
year	category

test_data dataset summary

	count	unique	top	freq	mean	\
absolute_humidity_2m:gm3	2160	106			8.206482	
air_density_2m:kgm3	2160	153			1.232807	
ceiling_height_agl:m	1473	1391			2938.389648	
clear_sky_energy_1h:J	2160	1807			1227746.75	
clear_sky_rad:W	2160	1044			341.056641	
cloud_base_agl:m	1879	1771			1797.160156	

dew_or_rime:idx	2160	3		0.040741
dew_point_2m:K	2160	202		280.783203
diffuse_rad:W	2160	985		84.915688
diffuse_rad_1h:J	2160	1806		305696.5
direct_rad:W	2160	916		114.279816
direct_rad_1h:J	2160	1634		411408.875
effective_cloud_cover:p	2160	590		64.113792
elevation:m	2160	3		12.333333
estimated_diff_hours	2160	24		27.5
fresh_snow_12h:cm	2160	2		0.000185
fresh_snow_1h:cm	2160	2		0.000185
fresh_snow_24h:cm	2160	2		0.000185
fresh_snow_3h:cm	2160	2		0.000185
fresh_snow_6h:cm	2160	2		0.000185
hour	2160	24		11.5
is_day:idx	2160	2		0.795833
is_in_shadow:idx	2160	2		0.24537
location	2160	3	A	720
month	2160	3		5.666667
msl_pressure:hPa	2160	321		1016.805786
precip_5min:mm	2160	27		0.00775
precip_type_5min:idx	2160	3		0.065741
pressure_100m:hPa	2160	359		1002.970825
pressure_50m:hPa	2160	356		1009.007202
prob_rime:p	2160	3		0.01588
rain_water:kgm2	2160	8		0.013056
relative_humidity_1000hPa:p	2160	538		70.920792
sfc_pressure:hPa	2160	363		1015.070374
snow_depth:cm	2160	1		0.0
snow_melt_10min:mm	2160	1		0.0
snow_water:kgm2	2160	16		0.060972
sun_azimuth:d	2160	1830		183.166199
sun_elevation:d	2160	1623		20.292332
super_cooled_liquid_water:kgm2	2160	7		0.065463
t_1000hPa:K	2160	254		284.737732
total_cloud_cover:p	2160	553		69.298981
visibility:m	2160	2155		33304.636719
weekend	2160	2		0.366667
wind_speed_10m:ms	2160	83		2.946759
wind_speed_u_10m:ms	2160	123		1.650694
wind_speed_v_10m:ms	2160	80		-0.187176
wind_speed_w_1000hPa:ms	2160	2		0.000324
year	2160	1		2023.0

	std	min	25%	\
absolute_humidity_2m:gm3	2.201396	3.2	6.6	
air_density_2m:kgm3	0.032116	1.142	1.209	
ceiling_height_agl:m	2913.641113	30.6	891.799988	

clear_sky_energy_1h:J	1104468.625	0.0	64338.124023	
clear_sky_rad:W	307.729095	0.0	13.65	
cloud_base_agl:m	2046.394409	29.799999	486.899994	
dew_or_rime:idx	0.202365	-1.0	0.0	
dew_point_2m:K	4.378817	268.0	277.899994	
diffuse_rad:W	78.422508	0.0	6.925	
diffuse_rad_1h:J	278146.25	0.0	36756.901367	
direct_rad:W	171.838226	0.0	0.0	
direct_rad_1h:J	611480.125	0.0	86.575001	
effective_cloud_cover:p	37.947498	0.0	30.700001	
elevation:m	8.261587	6.0	6.0	
estimated_diff_hours	6.923789	16.0	21.75	
fresh_snow_12h:cm	0.008607	0.0	0.0	
fresh_snow_1h:cm	0.008607	0.0	0.0	
fresh_snow_24h:cm	0.008607	0.0	0.0	
fresh_snow_3h:cm	0.008607	0.0	0.0	
fresh_snow_6h:cm	0.008607	0.0	0.0	
hour	6.923789	0.0	5.75	
is_day:idx	0.403185	0.0	1.0	
is_in_shadow:idx	0.430406	0.0	0.0	
location				
month	0.596423	5.0	5.0	
msl_pressure:hPa	9.728754	986.099976	1011.5	
precip_5min:mm	0.033776	0.0	0.0	
precip_type_5min:idx	0.249747	0.0	0.0	
pressure_100m:hPa	9.644145	971.799988	997.799988	
pressure_50m:hPa	9.74076	977.700012	1003.799988	
prob_rime:p	0.551282	0.0	0.0	
rain_water:kgm2	0.055256	0.0	0.0	
relative_humidity_1000hPa:p	15.725973	23.9	60.275	
sfc_pressure:hPa	9.840412	983.5	1009.799988	
snow_depth:cm	0.0	0.0	0.0	
snow_melt_10min:mm	0.0	-0.0	-0.0	
snow_water:kgm2	0.219562	0.0	0.0	
sun_azimuth:d	109.193207	8.27	85.359253	
sun_elevation:d	18.681047	-11.617	1.96475	
super_cooled_liquid_water:kgm2	0.115824	0.0	0.0	
t_1000hPa:K	5.839595	273.700012	279.799988	
total_cloud_cover:p	38.41222	0.0	32.799999	
visibility:m	15624.633789	874.400024	19635.100098	
weekend	0.482006	0.0	0.0	
wind_speed_10m:ms	1.733865	0.0	1.5	
wind_speed_u_10m:ms	2.578466	-4.3	-0.2	
wind_speed_v_10m:ms	1.50826	-4.4	-1.3	
wind_speed_w_1000hPa:ms	0.005685	-0.0	0.0	
year	0.0	2023.0	2023.0	

	50%	75%	max \

absolute_humidity_2m:gm3	8.0	10.0	14.2
air_density_2m:kgm3	1.238	1.26	1.301
ceiling_height_agl:m	1553.900024	4021.300049	11468.0
clear_sky_energy_1h:J	1056303.125	2372037.5	3005707.0
clear_sky_rad:W	273.849991	646.874985	835.099976
cloud_base_agl:m	997.799988	2298.300049	11467.799805
dew_or_rime:idx	0.0	0.0	1.0
dew_point_2m:K	281.0	284.299988	290.200012
diffuse_rad:W	73.700001	135.600006	312.600006
diffuse_rad_1h:J	272526.046875	488256.03125	1086246.25
direct_rad:W	16.200001	180.399994	668.0
direct_rad_1h:J	60416.199219	686746.859375	2403444.25
effective_cloud_cover:p	77.75	100.0	100.0
elevation:m	7.0	24.0	24.0
estimated_diff_hours	27.5	33.25	39.0
fresh_snow_12h:cm	0.0	0.0	0.4
fresh_snow_1h:cm	0.0	0.0	0.4
fresh_snow_24h:cm	0.0	0.0	0.4
fresh_snow_3h:cm	0.0	0.0	0.4
fresh_snow_6h:cm	0.0	0.0	0.4
hour	11.5	17.25	23.0
is_day:idx	1.0	1.0	1.0
is_in_shadow:idx	0.0	0.0	1.0
location			
month	6.0	6.0	7.0
msl_pressure:hPa	1020.599976	1023.799988	1029.599976
precip_5min:mm	0.0	0.0	0.34
precip_type_5min:idx	0.0	0.0	2.0
pressure_100m:hPa	1006.25	1010.099976	1016.400024
pressure_50m:hPa	1012.299988	1016.200012	1022.5
prob_rime:p	0.0	0.0	23.0
rain_water:kgm2	0.0	0.0	0.7
relative_humidity_1000hPa:p	73.900002	83.699997	98.900002
sfc_pressure:hPa	1018.299988	1022.299988	1028.699951
snow_depth:cm	0.0	0.0	0.0
snow_melt_10min:mm	0.0	0.0	0.0
snow_water:kgm2	0.0	0.0	3.4
sun_azimuth:d	184.236	279.576248	356.984009
sun_elevation:d	18.54	38.102499	49.902
super_cooled_liquid_water:kgm2	0.0	0.1	0.6
t_1000hPa:K	284.799988	288.299988	302.200012
total_cloud_cover:p	95.300003	100.0	100.0
visibility:m	37623.050781	45378.099609	63863.800781
weekend	0.0	1.0	1.0
wind_speed_10m:ms	2.7	4.0	8.8
wind_speed_u_10m:ms	1.6	3.525	8.8
wind_speed_v_10m:ms	-0.3	0.8	4.0
wind_speed_w_1000hPa:ms	0.0	0.0	0.1

					\
year	2023.0	2023.0	2023.0		
	dtypes	missing_count	missing_ratio	raw_type	
absolute_humidity_2m:gm3	float32			float	
air_density_2m:kgm3	float32			float	
ceiling_height_agl:m	float32	687	0.318056	float	
clear_sky_energy_1h:J	float32			float	
clear_sky_rad:W	float32			float	
cloud_base_agl:m	float32	281	0.130093	float	
dew_or_rime:idx	float32			float	
dew_point_2m:K	float32			float	
diffuse_rad:W	float32			float	
diffuse_rad_1h:J	float32			float	
direct_rad:W	float32			float	
direct_rad_1h:J	float32			float	
effective_cloud_cover:p	float32			float	
elevation:m	float32			float	
estimated_diff_hours	int64			int	
fresh_snow_12h:cm	float32			float	
fresh_snow_1h:cm	float32			float	
fresh_snow_24h:cm	float32			float	
fresh_snow_3h:cm	float32			float	
fresh_snow_6h:cm	float32			float	
hour	int64			int	
is_day:idx	float32			float	
is_in_shadow:idx	float32			float	
location	object			object	
month	int64			int	
msl_pressure:hPa	float32			float	
precip_5min:mm	float32			float	
precip_type_5min:idx	float32			float	
pressure_100m:hPa	float32			float	
pressure_50m:hPa	float32			float	
prob_rime:p	float32			float	
rain_water:kgm2	float32			float	
relative_humidity_1000hPa:p	float32			float	
sfc_pressure:hPa	float32			float	
snow_depth:cm	float32			float	
snow_melt_10min:mm	float32			float	
snow_water:kgm2	float32			float	
sun_azimuth:d	float32			float	
sun_elevation:d	float32			float	
super_cooled_liquid_water:kgm2	float32			float	
t_1000hPa:K	float32			float	
total_cloud_cover:p	float32			float	
visibility:m	float32			float	
weekend	int64			int	
wind_speed_10m:ms	float32			float	

wind_speed_u_10m:ms	float32	float
wind_speed_v_10m:ms	float32	float
wind_speed_w_1000hPa:ms	float32	float
year	int64	int

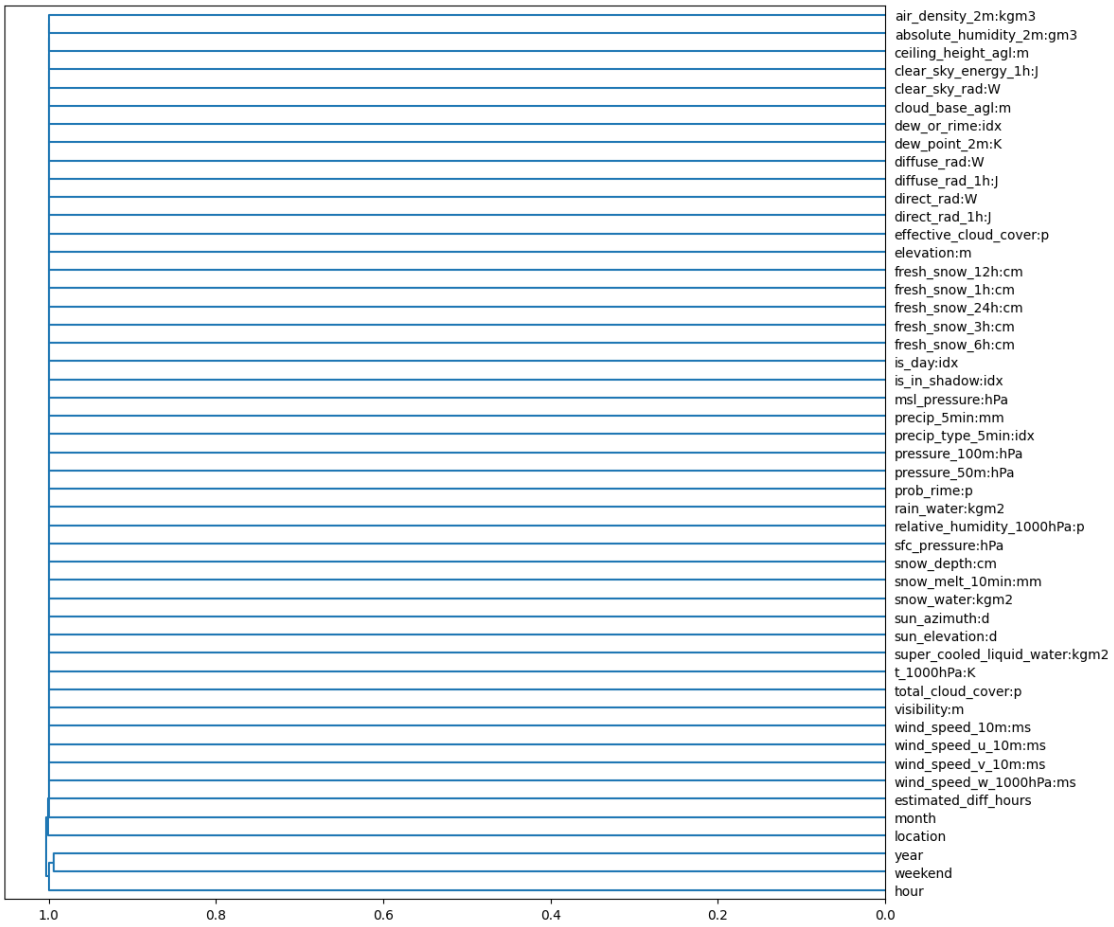
	variable_type	special_types
absolute_humidity_2m:gm3	numeric	
air_density_2m:kgm3	numeric	
ceiling_height_agl:m	numeric	
clear_sky_energy_1h:J	numeric	
clear_sky_rad:W	numeric	
cloud_base_agl:m	numeric	
dew_or_rime:idx	category	
dew_point_2m:K	numeric	
diffuse_rad:W	numeric	
diffuse_rad_1h:J	numeric	
direct_rad:W	numeric	
direct_rad_1h:J	numeric	
effective_cloud_cover:p	numeric	
elevation:m	category	
estimated_diff_hours	numeric	
fresh_snow_12h:cm	category	
fresh_snow_1h:cm	category	
fresh_snow_24h:cm	category	
fresh_snow_3h:cm	category	
fresh_snow_6h:cm	category	
hour	numeric	
is_day:idx	category	
is_in_shadow:idx	category	
location	category	
month	category	
msl_pressure:hPa	numeric	
precip_5min:mm	numeric	
precip_type_5min:idx	category	
pressure_100m:hPa	numeric	
pressure_50m:hPa	numeric	
prob_rime:p	category	
rain_water:kgm2	category	
relative_humidity_1000hPa:p	numeric	
sfc_pressure:hPa	numeric	
snow_depth:cm	category	
snow_melt_10min:mm	category	
snow_water:kgm2	category	
sun_azimuth:d	numeric	
sun_elevation:d	numeric	
super_cooled_liquid_water:kgm2	category	
t_1000hPa:K	numeric	
total_cloud_cover:p	numeric	

```
visibility:m                        numeric
weekend                            category
wind_speed_10m:ms                   numeric
wind_speed_u_10m:ms                 numeric
wind_speed_v_10m:ms                 numeric
wind_speed_w_1000hPa:ms            category
year                               category
```

Types warnings summary

```
                     train_data test_data warnings
estimated_diff_hours      float       int  warning
y                         float        --  warning
```

1.0.1 Feature Distance

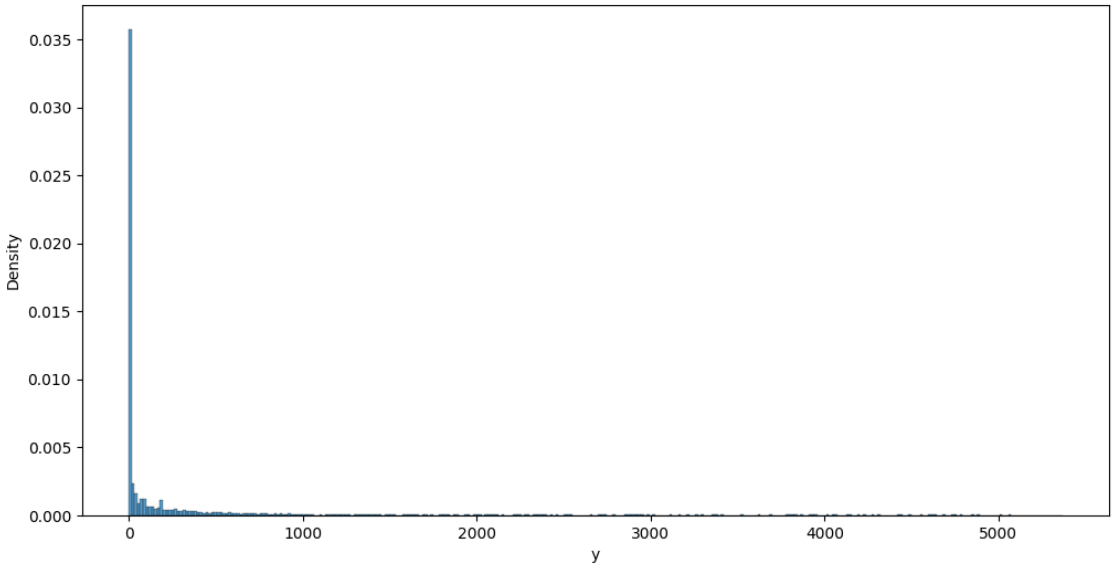


[4]:
```
auto.target_analysis(train_data=X_train, label="y")
```

1.1 Target variable analysis

	count	mean	std	min	25%	50%	75%	max	dtypes	\
y	10000	295.26029	787.46272	-0.0	0.0	0.0	176.4	5365.36	float64	

	unique	missing_count	missing_ratio	raw_type	special_types
y	2539			float	

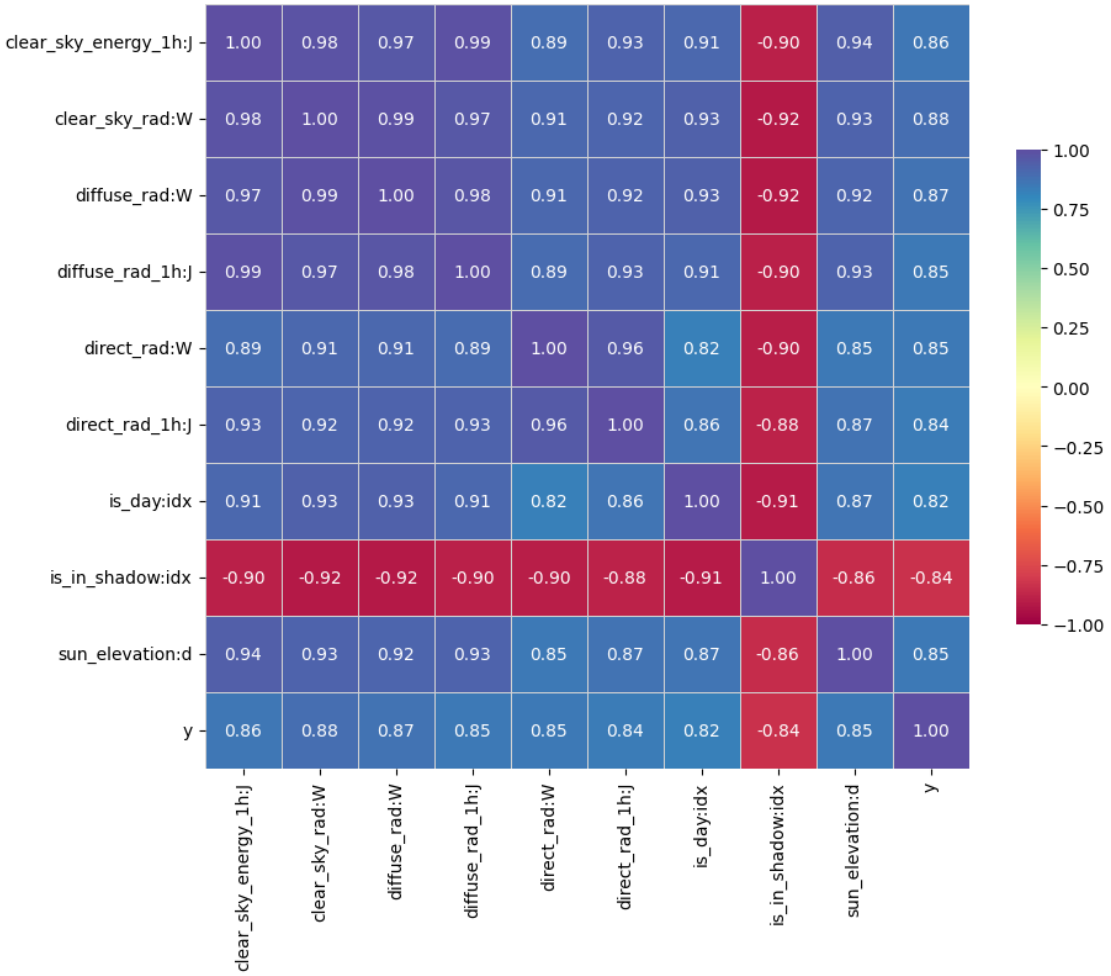


1.1.1 Distribution fits for target variable

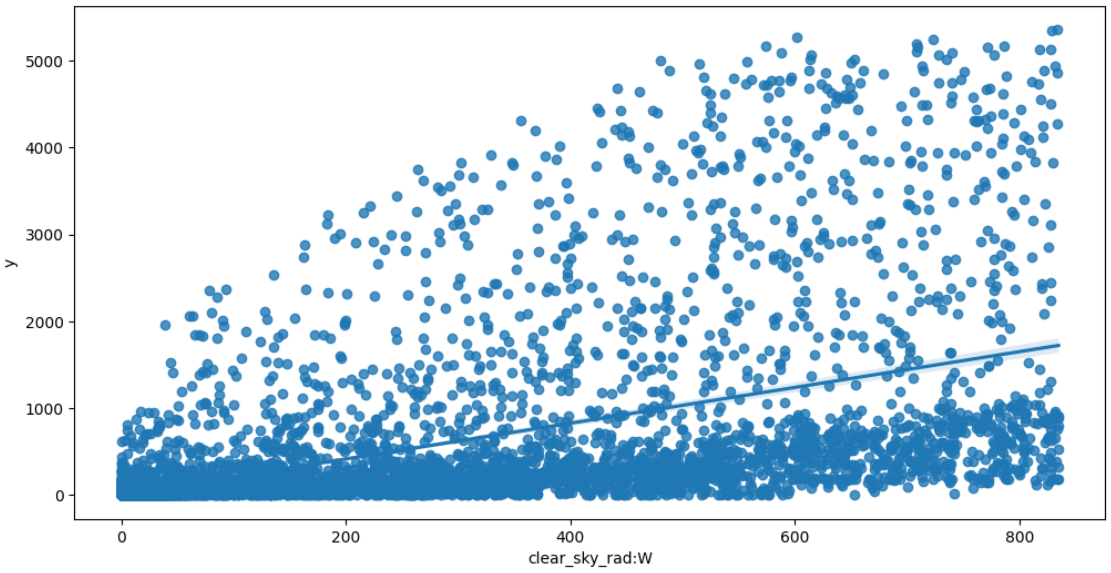
- none of the attempted distribution fits satisfy specified minimum p-value threshold: `0.01`

1.1.2 Target variable correlations

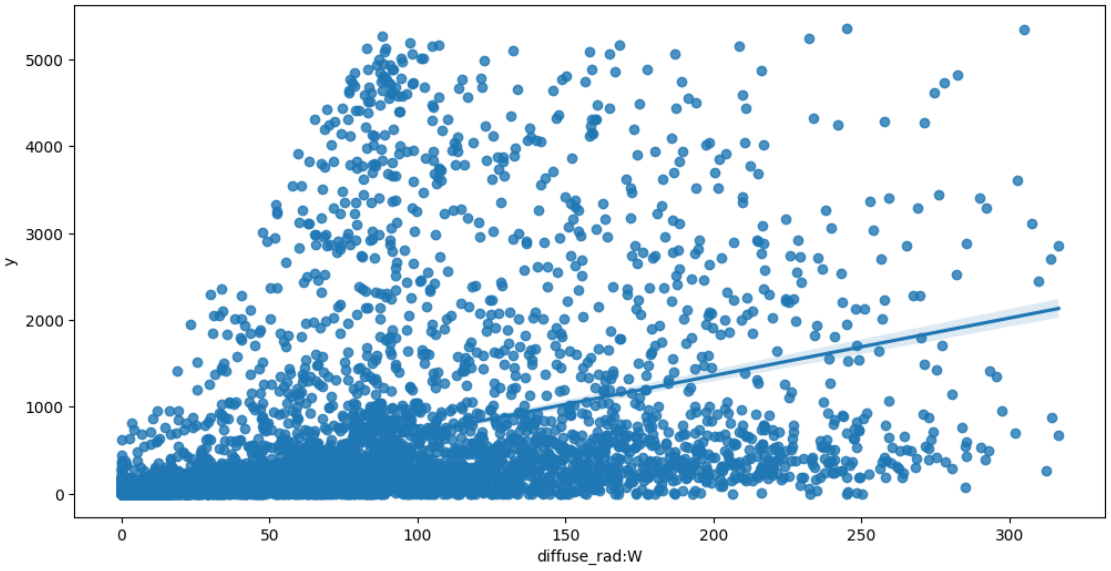
`train_data` - `spearman` correlation matrix; focus: absolute correlation for `y` >= `0.5` (sample size: 10000)



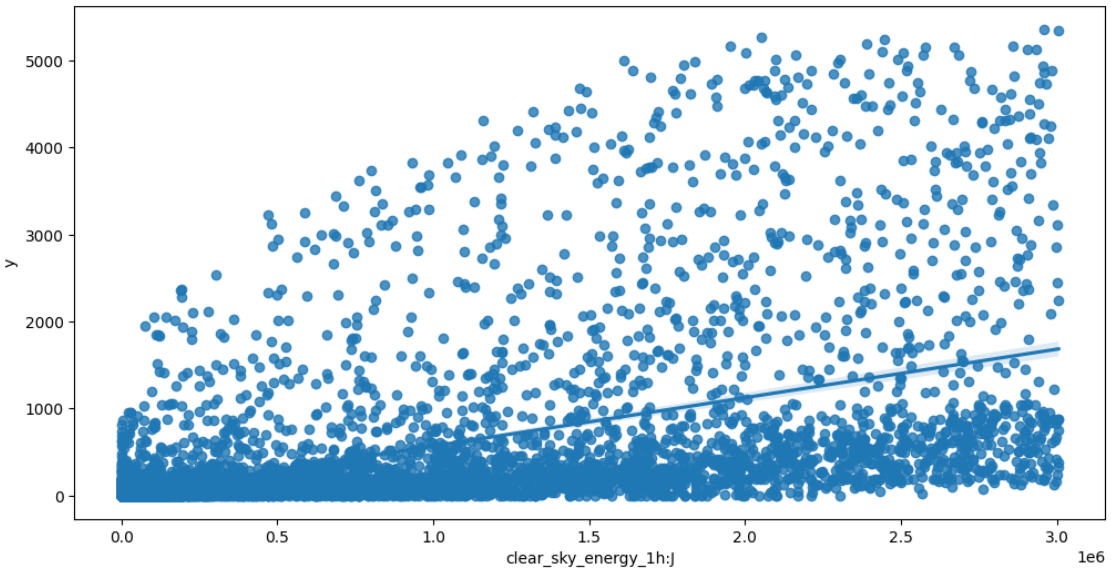
Feature interaction between `clear_sky_rad:W`/`y` in `train_data` (sample size: 10000)



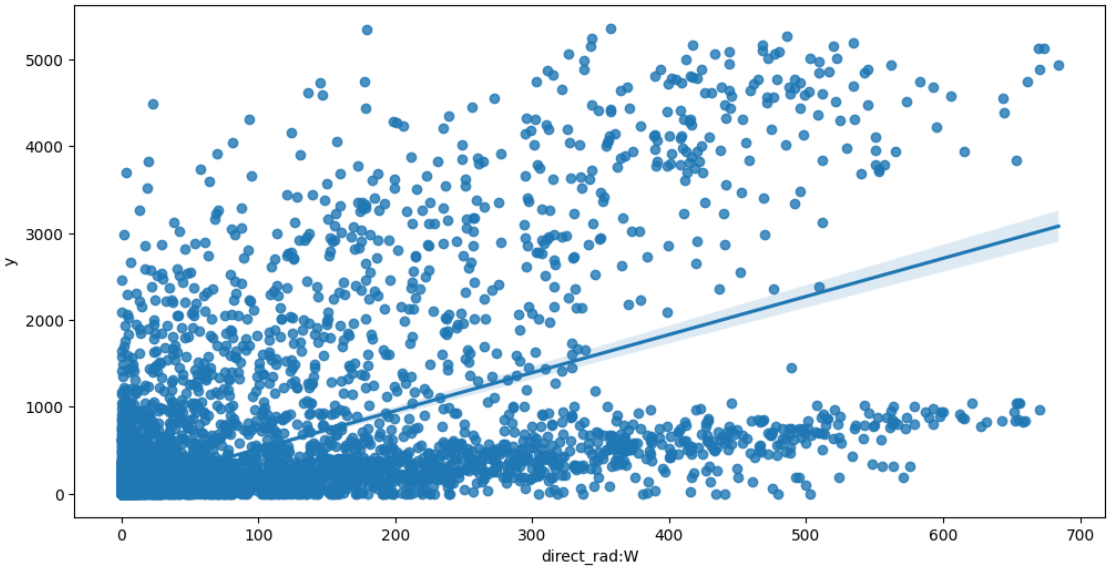
Feature interaction between `diffuse_rad:W/y` in `train_data` (sample size: 10000)



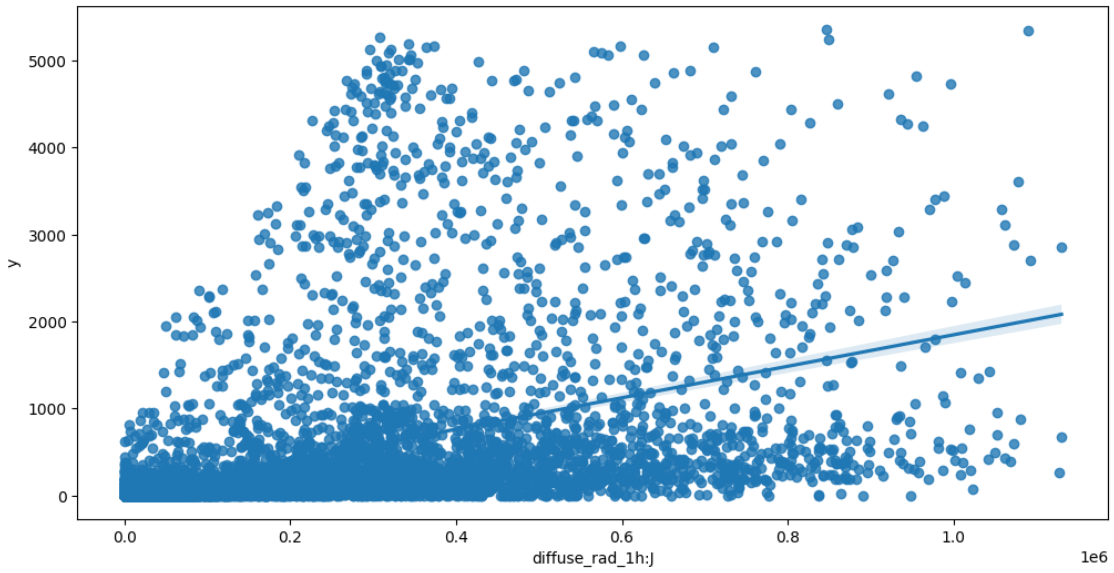
Feature interaction between `clear_sky_energy_1h:J/y` in `train_data` (sample size: 10000)



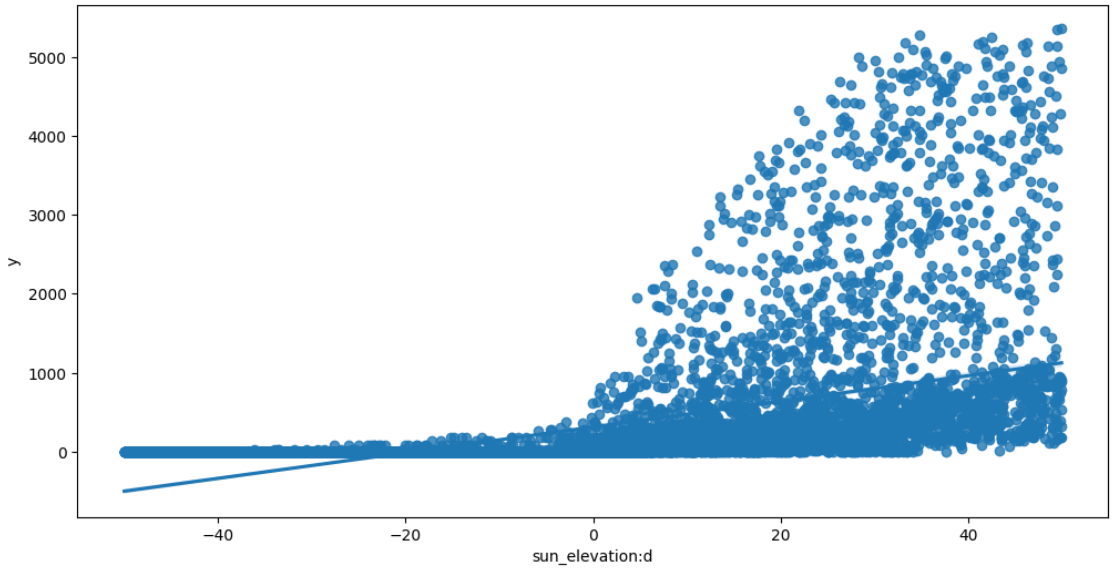
Feature interaction between `direct_rad:W/y` in `train_data` (sample size: 10000)



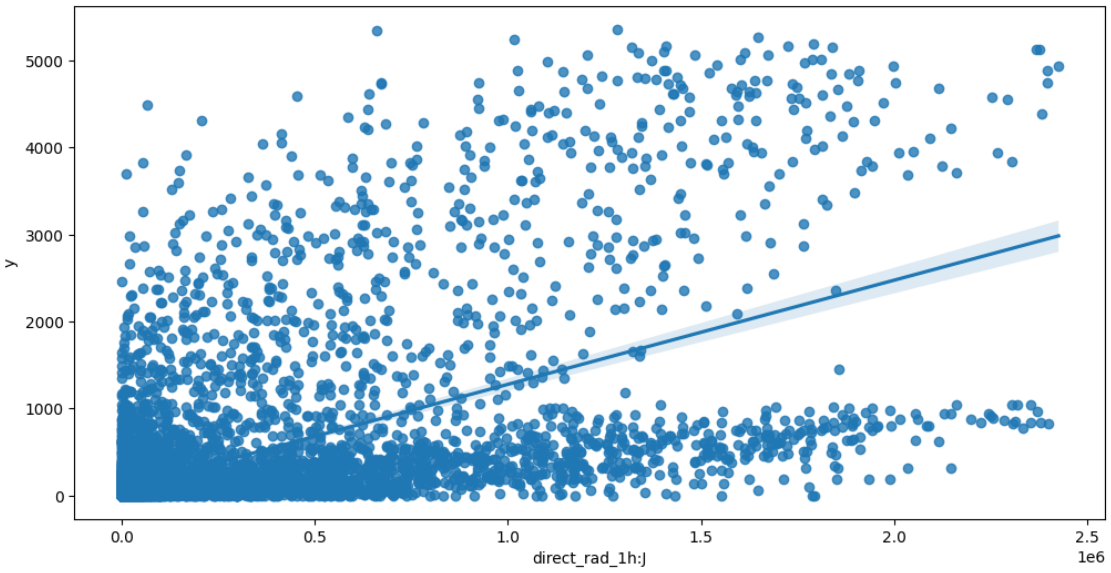
Feature interaction between `diffuse_rad_1h:J/y` in `train_data` (sample size: 10000)



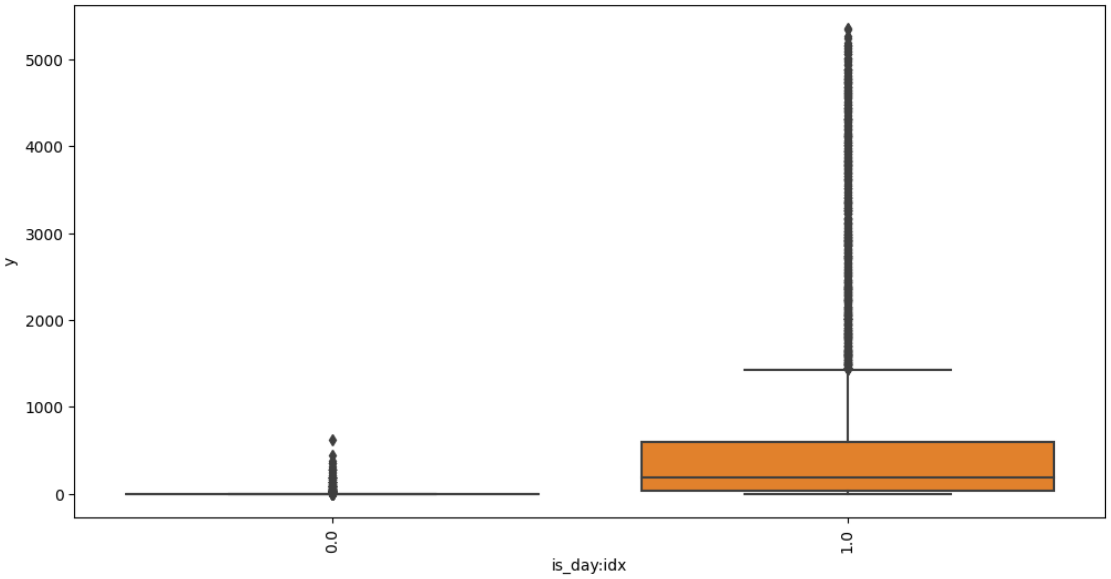
Feature interaction between `sun_elevation:d/y` in `train_data` (sample size: 10000)



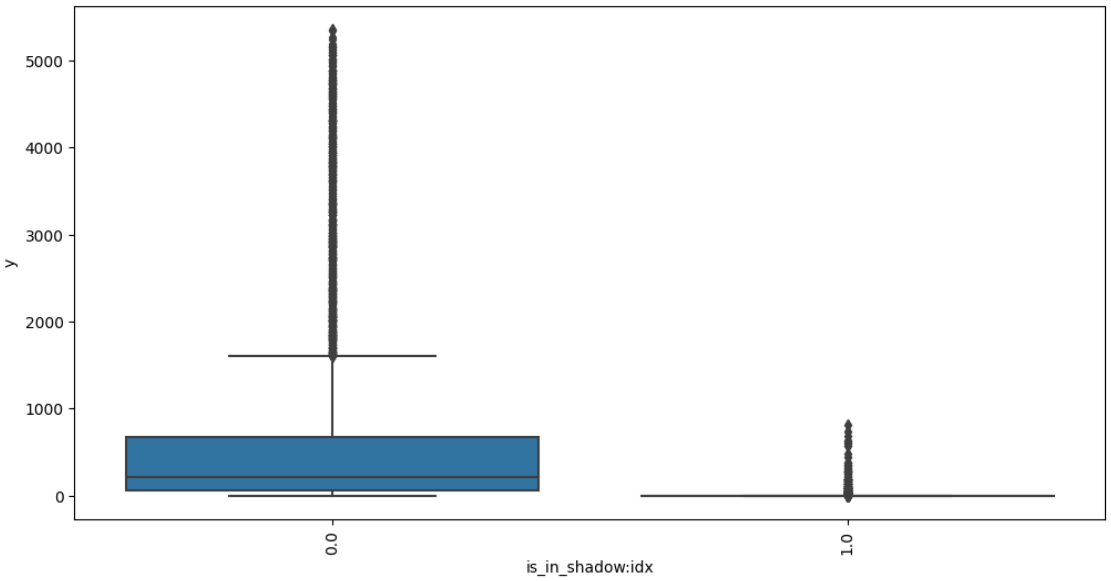
Feature interaction between `direct_rad_1h:J/y` in `train_data` (sample size: 10000)



Feature interaction between `is_day:idx/y` in `train_data` (sample size: 10000)



Feature interaction between `is_in_shadow:idx/y` in `train_data` (sample size: 10000)



2 Starting

[5]:
```
import os


# Get the last submission number
last_submission_number = int(max([int(filename.split('_')[1].split('.')[0]) for␣
 ↪filename in os.listdir('submissions') if "submission" in filename]))
print("Last submission number:", last_submission_number)
print("Now creating submission number:", last_submission_number + 1)

# Create the new filename
new_filename = f'submission_{last_submission_number + 1}'

hello = os.environ.get('HELLO')
if hello is not None:
    new_filename += f'_{hello}'

print("New filename:", new_filename)
```

```
Last submission number: 80
Now creating submission number: 81
New filename: submission_81_jorge
```

[6]:
```
from autogluon.tabular import TabularDataset, TabularPredictor
train_data = TabularDataset('X_train_raw.csv')
train_data.drop(columns=['ds'], inplace=True)
```

```
label = 'y'
metric = 'mean_absolute_error'
time_limit = 60
presets = 'best_quality'
```

[7]:
```
predictors = [None, None, None]
```

[8]:
```
loc = "A"
print(f"Training model for location {loc}...")
predictor = TabularPredictor(label=label, eval_metric=metric,␣
 ↪path=f"AutogluonModels/{new_filename}_{loc}").
 ↪fit(train_data[train_data["location"] == loc], time_limit=time_limit,␣
 ↪presets=presets)
predictors[0] = predictor
```

```
Presets specified: ['best_quality']
Stack configuration (auto_stack=True): num_stack_levels=1, num_bag_folds=8,
num_bag_sets=20
Beginning AutoGluon training … Time limit = 60s
AutoGluon will save models to "AutogluonModels/submission_81_jorge_A/"
AutoGluon Version:  0.8.1
Python Version:     3.10.12
Operating System:   Darwin
Platform Machine:   arm64
Platform Version:   Darwin Kernel Version 22.1.0: Sun Oct  9 20:15:09 PDT 2022;
root:xnu-8792.41.9~2/RELEASE_ARM64_T6000
Disk Space Avail:   30.19 GB / 494.38 GB (6.1%)
Train Data Rows:    34085
Train Data Columns: 49
Label Column: y
Preprocessing data …
AutoGluon infers your prediction problem is: 'regression' (because dtype of
label-column == float and many unique label-values observed).
        Label info (max, min, mean, stddev): (5733.42, 0.0, 630.59471,
1165.90242)
        If 'regression' is not the correct problem_type, please manually specify
the problem_type parameter during predictor init (You may specify problem_type
as one of: ['binary', 'multiclass', 'regression'])
Using Feature Generators to preprocess the data …
Fitting AutoMLPipelineFeatureGenerator…
        Available Memory:                    4929.97 MB
        Train Data (Original)  Memory Usage: 15.07 MB (0.3% of available memory)
        Inferring data type of each feature based on column values. Set
feature_metadata_in to manually specify special dtypes of the features.
        Stage 1 Generators:
                Fitting AsTypeFeatureGenerator…
                        Note: Converting 1 features to boolean dtype as they
```

```
only contain 2 unique values.
        Stage 2 Generators:
                Fitting FillNaFeatureGenerator…
        Stage 3 Generators:
                Fitting IdentityFeatureGenerator…
        Stage 4 Generators:
                Fitting DropUniqueFeatureGenerator…
        Stage 5 Generators:
                Fitting DropDuplicatesFeatureGenerator…
        Useless Original Features (Count: 1): ['location']
                These features carry no predictive signal and should be manually
investigated.
                This is typically a feature which has the same value for all
rows.
                These features do not need to be present at inference time.
        Types of features in original data (raw dtype, special dtypes):
                ('float', []) : 44 | ['absolute_humidity_2m:gm3',
'air_density_2m:kgm3', 'ceiling_height_agl:m', 'clear_sky_energy_1h:J',
'clear_sky_rad:W', …]
                ('int', [])   :  4 | ['hour', 'weekday', 'month', 'year']
        Types of features in processed data (raw dtype, special dtypes):
                ('float', [])     : 43 | ['absolute_humidity_2m:gm3',
'air_density_2m:kgm3', 'ceiling_height_agl:m', 'clear_sky_energy_1h:J',
'clear_sky_rad:W', …]
                ('int', [])       :  4 | ['hour', 'weekday', 'month', 'year']
                ('int', ['bool']) :  1 | ['elevation:m']
        0.1s = Fit runtime
        48 features in original data used to generate 48 features in processed
data.
        Train Data (Processed) Memory Usage: 12.85 MB (0.3% of available memory)
Data preprocessing and feature engineering runtime = 0.13s …
AutoGluon will gauge predictive performance using evaluation metric:
'mean_absolute_error'
        This metric's sign has been flipped to adhere to being higher_is_better.
The metric score can be multiplied by -1 to get the metric value.
        To change this, specify the eval_metric parameter of Predictor()
User-specified model hyperparameters to be fit:
{
        'NN_TORCH': {},
        'GBM': [{'extra_trees': True, 'ag_args': {'name_suffix': 'XT'}}, {},
'GBMLarge'],
        'CAT': {},
        'XGB': {},
        'FASTAI': {},
        'RF': [{'criterion': 'gini', 'ag_args': {'name_suffix': 'Gini',
'problem_types': ['binary', 'multiclass']}}, {'criterion': 'entropy', 'ag_args':
{'name_suffix': 'Entr', 'problem_types': ['binary', 'multiclass']}},
{'criterion': 'squared_error', 'ag_args': {'name_suffix': 'MSE',
```

```
'problem_types': ['regression', 'quantile']}}],
        'XT': [{'criterion': 'gini', 'ag_args': {'name_suffix': 'Gini',
'problem_types': ['binary', 'multiclass']}}, {'criterion': 'entropy', 'ag_args':
{'name_suffix': 'Entr', 'problem_types': ['binary', 'multiclass']}},
{'criterion': 'squared_error', 'ag_args': {'name_suffix': 'MSE',
'problem_types': ['regression', 'quantile']}}],
        'KNN': [{'weights': 'uniform', 'ag_args': {'name_suffix': 'Unif'}},
{'weights': 'distance', 'ag_args': {'name_suffix': 'Dist'}}],
}
AutoGluon will fit 2 stack levels (L1 to L2) …
Fitting 11 L1 models …
Fitting model: KNeighborsUnif_BAG_L1 … Training model for up to 39.91s of the
59.87s of remaining time.

Training model for location A…

        Not enough time to generate out-of-fold predictions for model. Estimated
time required was 170.91s compared to 51.85s of available time.
        Time limit exceeded… Skipping KNeighborsUnif_BAG_L1.
Fitting model: KNeighborsDist_BAG_L1 … Training model for up to 37.36s of the
57.32s of remaining time.
        Not enough time to generate out-of-fold predictions for model. Estimated
time required was 205.93s compared to 48.54s of available time.
        Time limit exceeded… Skipping KNeighborsDist_BAG_L1.
Fitting model: LightGBMXT_BAG_L1 … Training model for up to 34.29s of the
54.26s of remaining time.
        Fitting 8 child models (S1F1 - S1F8) | Fitting with
ParallelLocalFoldFittingStrategy
        -161.5377       = Validation score   (-mean_absolute_error)
        27.12s   = Training   runtime
        61.97s   = Validation runtime
Completed 1/20 k-fold bagging repeats …
Fitting model: WeightedEnsemble_L2 … Training model for up to 59.87s of the
13.14s of remaining time.
        -161.5377       = Validation score   (-mean_absolute_error)
        0.01s    = Training   runtime
        0.0s     = Validation runtime
Fitting 9 L2 models …
Fitting model: LightGBMXT_BAG_L2 … Training model for up to 13.13s of the
13.12s of remaining time.
        Fitting 8 child models (S1F1 - S1F8) | Fitting with
ParallelLocalFoldFittingStrategy
        -163.4182       = Validation score   (-mean_absolute_error)
        2.35s    = Training   runtime
        0.53s    = Validation runtime
Fitting model: LightGBM_BAG_L2 … Training model for up to 8.54s of the 8.53s
of remaining time.
        Fitting 8 child models (S1F1 - S1F8) | Fitting with
ParallelLocalFoldFittingStrategy
```

```
        -161.3785       = Validation score   (-mean_absolute_error)
        1.81s    = Training   runtime
        0.16s    = Validation runtime
Fitting model: RandomForestMSE_BAG_L2 … Training model for up to 4.8s of the
4.8s of remaining time.
        -160.8174       = Validation score   (-mean_absolute_error)
        24.86s   = Training   runtime
        0.86s    = Validation runtime
Completed 1/20 k-fold bagging repeats …
Fitting model: WeightedEnsemble_L3 … Training model for up to 59.87s of the
-21.27s of remaining time.
        -159.4455       = Validation score   (-mean_absolute_error)
        0.16s    = Training   runtime
        0.0s     = Validation runtime
AutoGluon training complete, total runtime = 81.45s … Best model:
"WeightedEnsemble_L3"
TabularPredictor saved. To load, use: predictor =
TabularPredictor.load("AutogluonModels/submission_81_jorge_A/")
```

[9]:
```
loc = "B"
print(f"Training model for location {loc}...")
predictor = TabularPredictor(label=label, eval_metric=metric,␣
 ↪path=f"AutogluonModels/{new_filename}_{loc}").
 ↪fit(train_data[train_data["location"] == loc], time_limit=time_limit,␣
 ↪presets=presets)
predictors[1] = predictor
```

```
Presets specified: ['best_quality']
Stack configuration (auto_stack=True): num_stack_levels=1, num_bag_folds=8,
num_bag_sets=20
Beginning AutoGluon training … Time limit = 60s
AutoGluon will save models to "AutogluonModels/submission_81_jorge_B/"
AutoGluon Version:  0.8.1
Python Version:     3.10.12
Operating System:   Darwin
Platform Machine:   arm64
Platform Version:   Darwin Kernel Version 22.1.0: Sun Oct  9 20:15:09 PDT 2022;
root:xnu-8792.41.9~2/RELEASE_ARM64_T6000
Disk Space Avail:   29.45 GB / 494.38 GB (6.0%)
Train Data Rows:    32844
Train Data Columns: 49
Label Column: y
Preprocessing data …
AutoGluon infers your prediction problem is: 'regression' (because dtype of
label-column == float and many unique label-values observed).
        Label info (max, min, mean, stddev): (1152.3, -0.0, 96.82478, 193.94649)
        If 'regression' is not the correct problem_type, please manually specify
the problem_type parameter during predictor init (You may specify problem_type
```

```
as one of: ['binary', 'multiclass', 'regression'])
Using Feature Generators to preprocess the data …
Fitting AutoMLPipelineFeatureGenerator…
        Available Memory:                    4331.9 MB
        Train Data (Original)  Memory Usage: 14.52 MB (0.3% of available memory)
        Inferring data type of each feature based on column values. Set
feature_metadata_in to manually specify special dtypes of the features.
        Stage 1 Generators:
                Fitting AsTypeFeatureGenerator…
                        Note: Converting 1 features to boolean dtype as they
only contain 2 unique values.
        Stage 2 Generators:
                Fitting FillNaFeatureGenerator…
        Stage 3 Generators:
                Fitting IdentityFeatureGenerator…
        Stage 4 Generators:
                Fitting DropUniqueFeatureGenerator…
        Stage 5 Generators:
                Fitting DropDuplicatesFeatureGenerator…
        Useless Original Features (Count: 1): ['location']
                These features carry no predictive signal and should be manually
investigated.
                This is typically a feature which has the same value for all
rows.
                These features do not need to be present at inference time.
        Types of features in original data (raw dtype, special dtypes):
                ('float', []) : 44 | ['absolute_humidity_2m:gm3',
'air_density_2m:kgm3', 'ceiling_height_agl:m', 'clear_sky_energy_1h:J',
'clear_sky_rad:W', …]
                ('int', [])   :  4 | ['hour', 'weekday', 'month', 'year']
        Types of features in processed data (raw dtype, special dtypes):
                ('float', [])     : 43 | ['absolute_humidity_2m:gm3',
'air_density_2m:kgm3', 'ceiling_height_agl:m', 'clear_sky_energy_1h:J',
'clear_sky_rad:W', …]
                ('int', [])       :  4 | ['hour', 'weekday', 'month', 'year']
                ('int', ['bool']) :  1 | ['elevation:m']
        0.1s = Fit runtime
        48 features in original data used to generate 48 features in processed
data.
        Train Data (Processed) Memory Usage: 12.38 MB (0.3% of available memory)
Data preprocessing and feature engineering runtime = 0.12s …
AutoGluon will gauge predictive performance using evaluation metric:
'mean_absolute_error'
        This metric's sign has been flipped to adhere to being higher_is_better.
The metric score can be multiplied by -1 to get the metric value.
        To change this, specify the eval_metric parameter of Predictor()
User-specified model hyperparameters to be fit:
{
```

```
        'NN_TORCH': {},
        'GBM': [{'extra_trees': True, 'ag_args': {'name_suffix': 'XT'}}, {},
'GBMLarge'],
        'CAT': {},
        'XGB': {},
        'FASTAI': {},
        'RF': [{'criterion': 'gini', 'ag_args': {'name_suffix': 'Gini',
'problem_types': ['binary', 'multiclass']}}, {'criterion': 'entropy', 'ag_args':
{'name_suffix': 'Entr', 'problem_types': ['binary', 'multiclass']}},
{'criterion': 'squared_error', 'ag_args': {'name_suffix': 'MSE',
'problem_types': ['regression', 'quantile']}}],
        'XT': [{'criterion': 'gini', 'ag_args': {'name_suffix': 'Gini',
'problem_types': ['binary', 'multiclass']}}, {'criterion': 'entropy', 'ag_args':
{'name_suffix': 'Entr', 'problem_types': ['binary', 'multiclass']}},
{'criterion': 'squared_error', 'ag_args': {'name_suffix': 'MSE',
'problem_types': ['regression', 'quantile']}}],
        'KNN': [{'weights': 'uniform', 'ag_args': {'name_suffix': 'Unif'}},
{'weights': 'distance', 'ag_args': {'name_suffix': 'Dist'}}],
}
AutoGluon will fit 2 stack levels (L1 to L2) …
Fitting 11 L1 models …
Fitting model: KNeighborsUnif_BAG_L1 … Training model for up to 39.91s of the
59.88s of remaining time.

Training model for location B…

        Not enough time to generate out-of-fold predictions for model. Estimated
time required was 195.85s compared to 51.85s of available time.
        Time limit exceeded… Skipping KNeighborsUnif_BAG_L1.
Fitting model: KNeighborsDist_BAG_L1 … Training model for up to 36.88s of the
56.85s of remaining time.
        Not enough time to generate out-of-fold predictions for model. Estimated
time required was 220.95s compared to 47.91s of available time.
        Time limit exceeded… Skipping KNeighborsDist_BAG_L1.
Fitting model: LightGBMXT_BAG_L1 … Training model for up to 33.46s of the
53.43s of remaining time.
        Fitting 8 child models (S1F1 - S1F8) | Fitting with
ParallelLocalFoldFittingStrategy
        -25.7449         = Validation score   (-mean_absolute_error)
        28.32s   = Training   runtime
        65.73s   = Validation runtime
Completed 1/20 k-fold bagging repeats …
Fitting model: WeightedEnsemble_L2 … Training model for up to 59.88s of the
14.16s of remaining time.
        -25.7449         = Validation score   (-mean_absolute_error)
        0.01s    = Training   runtime
        0.0s     = Validation runtime
Fitting 9 L2 models …
Fitting model: LightGBMXT_BAG_L2 … Training model for up to 14.1s of the
```

```
14.06s of remaining time.
        Fitting 8 child models (S1F1 - S1F8) | Fitting with
ParallelLocalFoldFittingStrategy
        -24.2516        = Validation score   (-mean_absolute_error)
        5.29s    = Training   runtime
        1.3s     = Validation runtime
Fitting model: LightGBM_BAG_L2 … Training model for up to 4.97s of the 4.94s
of remaining time.
        Fitting 8 child models (S1F1 - S1F8) | Fitting with
ParallelLocalFoldFittingStrategy
        -23.6458        = Validation score   (-mean_absolute_error)
        1.99s    = Training   runtime
        0.19s    = Validation runtime
Fitting model: RandomForestMSE_BAG_L2 … Training model for up to 0.91s of the
0.89s of remaining time.
        -22.1865        = Validation score   (-mean_absolute_error)
        25.62s   = Training   runtime
        0.85s    = Validation runtime
Completed 1/20 k-fold bagging repeats …
Fitting model: WeightedEnsemble_L3 … Training model for up to 59.88s of the
-25.95s of remaining time.
        -22.1865        = Validation score   (-mean_absolute_error)
        0.14s    = Training   runtime
        0.0s     = Validation runtime
AutoGluon training complete, total runtime = 86.12s … Best model:
"WeightedEnsemble_L3"
TabularPredictor saved. To load, use: predictor =
TabularPredictor.load("AutogluonModels/submission_81_jorge_B/")
```

[10]:
```
loc = "C"
print(f"Training model for location {loc}...")
predictor = TabularPredictor(label=label, eval_metric=metric,␣
 ↪path=f"AutogluonModels/{new_filename}_{loc}").
 ↪fit(train_data[train_data["location"] == loc], time_limit=time_limit,␣
 ↪presets=presets)
predictors[2] = predictor
```

```
Presets specified: ['best_quality']
Stack configuration (auto_stack=True): num_stack_levels=1, num_bag_folds=8,
num_bag_sets=20
Beginning AutoGluon training … Time limit = 60s
AutoGluon will save models to "AutogluonModels/submission_81_jorge_C/"
AutoGluon Version:  0.8.1
Python Version:     3.10.12
Operating System:   Darwin
Platform Machine:   arm64
Platform Version:   Darwin Kernel Version 22.1.0: Sun Oct  9 20:15:09 PDT 2022;
root:xnu-8792.41.9~2/RELEASE_ARM64_T6000
```

```
Disk Space Avail:   28.83 GB / 494.38 GB (5.8%)
Train Data Rows:    26095
Train Data Columns: 49
Label Column: y
Preprocessing data …
AutoGluon infers your prediction problem is: 'regression' (because dtype of
label-column == float and label-values can't be converted to int).
        Label info (max, min, mean, stddev): (999.6, -0.0, 77.63106, 165.81688)
        If 'regression' is not the correct problem_type, please manually specify
the problem_type parameter during predictor init (You may specify problem_type
as one of: ['binary', 'multiclass', 'regression'])
Using Feature Generators to preprocess the data …
Fitting AutoMLPipelineFeatureGenerator…
        Available Memory:                    4428.79 MB
        Train Data (Original)  Memory Usage: 11.53 MB (0.3% of available memory)
        Inferring data type of each feature based on column values. Set
feature_metadata_in to manually specify special dtypes of the features.
        Stage 1 Generators:
                Fitting AsTypeFeatureGenerator…
                        Note: Converting 1 features to boolean dtype as they
only contain 2 unique values.
        Stage 2 Generators:
                Fitting FillNaFeatureGenerator…
        Stage 3 Generators:
                Fitting IdentityFeatureGenerator…
        Stage 4 Generators:
                Fitting DropUniqueFeatureGenerator…
        Stage 5 Generators:
                Fitting DropDuplicatesFeatureGenerator…
        Useless Original Features (Count: 1): ['location']
                These features carry no predictive signal and should be manually
investigated.
                This is typically a feature which has the same value for all
rows.
                These features do not need to be present at inference time.
        Types of features in original data (raw dtype, special dtypes):
                ('float', []) : 44 | ['absolute_humidity_2m:gm3',
'air_density_2m:kgm3', 'ceiling_height_agl:m', 'clear_sky_energy_1h:J',
'clear_sky_rad:W', …]
                ('int', [])   :  4 | ['hour', 'weekday', 'month', 'year']
        Types of features in processed data (raw dtype, special dtypes):
                ('float', [])     : 43 | ['absolute_humidity_2m:gm3',
'air_density_2m:kgm3', 'ceiling_height_agl:m', 'clear_sky_energy_1h:J',
'clear_sky_rad:W', …]
                ('int', [])       :  4 | ['hour', 'weekday', 'month', 'year']
                ('int', ['bool']) :  1 | ['elevation:m']
        0.1s = Fit runtime
        48 features in original data used to generate 48 features in processed
```

```
data.
        Train Data (Processed) Memory Usage: 9.84 MB (0.2% of available memory)
Data preprocessing and feature engineering runtime = 0.13s …
AutoGluon will gauge predictive performance using evaluation metric:
'mean_absolute_error'
        This metric's sign has been flipped to adhere to being higher_is_better.
The metric score can be multiplied by -1 to get the metric value.
        To change this, specify the eval_metric parameter of Predictor()
User-specified model hyperparameters to be fit:
{
        'NN_TORCH': {},
        'GBM': [{'extra_trees': True, 'ag_args': {'name_suffix': 'XT'}}, {},
'GBMLarge'],
        'CAT': {},
        'XGB': {},
        'FASTAI': {},
        'RF': [{'criterion': 'gini', 'ag_args': {'name_suffix': 'Gini',
'problem_types': ['binary', 'multiclass']}}, {'criterion': 'entropy', 'ag_args':
{'name_suffix': 'Entr', 'problem_types': ['binary', 'multiclass']}},
{'criterion': 'squared_error', 'ag_args': {'name_suffix': 'MSE',
'problem_types': ['regression', 'quantile']}}],
        'XT': [{'criterion': 'gini', 'ag_args': {'name_suffix': 'Gini',
'problem_types': ['binary', 'multiclass']}}, {'criterion': 'entropy', 'ag_args':
{'name_suffix': 'Entr', 'problem_types': ['binary', 'multiclass']}},
{'criterion': 'squared_error', 'ag_args': {'name_suffix': 'MSE',
'problem_types': ['regression', 'quantile']}}],
        'KNN': [{'weights': 'uniform', 'ag_args': {'name_suffix': 'Unif'}},
{'weights': 'distance', 'ag_args': {'name_suffix': 'Dist'}}],
}
AutoGluon will fit 2 stack levels (L1 to L2) …
Fitting 11 L1 models …
Fitting model: KNeighborsUnif_BAG_L1 … Training model for up to 39.9s of the
59.87s of remaining time.

Training model for location C…

        Not enough time to generate out-of-fold predictions for model. Estimated
time required was 99.89s compared to 51.85s of available time.
        Time limit exceeded… Skipping KNeighborsUnif_BAG_L1.
Fitting model: KNeighborsDist_BAG_L1 … Training model for up to 37.94s of the
57.9s of remaining time.
        Not enough time to generate out-of-fold predictions for model. Estimated
time required was 86.14s compared to 49.3s of available time.
        Time limit exceeded… Skipping KNeighborsDist_BAG_L1.
Fitting model: LightGBMXT_BAG_L1 … Training model for up to 36.25s of the
56.21s of remaining time.
        Fitting 8 child models (S1F1 - S1F8) | Fitting with
ParallelLocalFoldFittingStrategy
        -17.0996         = Validation score   (-mean_absolute_error)
```

```
        23.53s   = Training   runtime
        44.64s   = Validation runtime
Fitting model: LightGBM_BAG_L1 … Training model for up to 4.44s of the 24.41s
of remaining time.
        Fitting 8 child models (S1F1 - S1F8) | Fitting with
ParallelLocalFoldFittingStrategy
        -19.0788         = Validation score   (-mean_absolute_error)
        4.37s    = Training   runtime
        1.22s    = Validation runtime
Completed 1/20 k-fold bagging repeats …
Fitting model: WeightedEnsemble_L2 … Training model for up to 59.87s of the
17.79s of remaining time.
        -17.0784         = Validation score   (-mean_absolute_error)
        0.09s    = Training   runtime
        0.0s     = Validation runtime
Fitting 9 L2 models …
Fitting model: LightGBMXT_BAG_L2 … Training model for up to 17.69s of the
17.68s of remaining time.
        Fitting 8 child models (S1F1 - S1F8) | Fitting with
ParallelLocalFoldFittingStrategy
        -17.7999         = Validation score   (-mean_absolute_error)
        2.08s    = Training   runtime
        0.4s     = Validation runtime
Fitting model: LightGBM_BAG_L2 … Training model for up to 13.48s of the 13.48s
of remaining time.
        Fitting 8 child models (S1F1 - S1F8) | Fitting with
ParallelLocalFoldFittingStrategy
        -17.2792         = Validation score   (-mean_absolute_error)
        1.61s    = Training   runtime
        0.12s    = Validation runtime
Fitting model: RandomForestMSE_BAG_L2 … Training model for up to 10.02s of the
10.01s of remaining time.
        -17.0599         = Validation score   (-mean_absolute_error)
        19.24s   = Training   runtime
        0.52s    = Validation runtime
Completed 1/20 k-fold bagging repeats …
Fitting model: WeightedEnsemble_L3 … Training model for up to 59.87s of the
-9.96s of remaining time.
        -16.9596         = Validation score   (-mean_absolute_error)
        0.14s    = Training   runtime
        0.0s     = Validation runtime
AutoGluon training complete, total runtime = 70.12s … Best model:
"WeightedEnsemble_L3"
TabularPredictor saved. To load, use: predictor =
TabularPredictor.load("AutogluonModels/submission_81_jorge_C/")
```

3 Submit

[11]:
```
import pandas as pd
import matplotlib.pyplot as plt

train_data_with_dates = TabularDataset('X_train_raw.csv')
train_data_with_dates["ds"] = pd.to_datetime(train_data_with_dates["ds"])

test_data = TabularDataset('X_test_raw.csv')
test_data["ds"] = pd.to_datetime(test_data["ds"])
#test_data
```

```
Loaded data from: X_train_raw.csv | Columns = 51 / 51 | Rows = 93024 -> 93024
Loaded data from: X_test_raw.csv | Columns = 50 / 50 | Rows = 2160 -> 2160
```

[12]:
```
test_ids = TabularDataset('test.csv')
test_ids["time"] = pd.to_datetime(test_ids["time"])
# merge test_data with test_ids
test_data_merged = pd.merge(test_data, test_ids, how="inner", right_on=["time",␣
 ↪"location"], left_on=["ds", "location"])

#test_data_merged
```

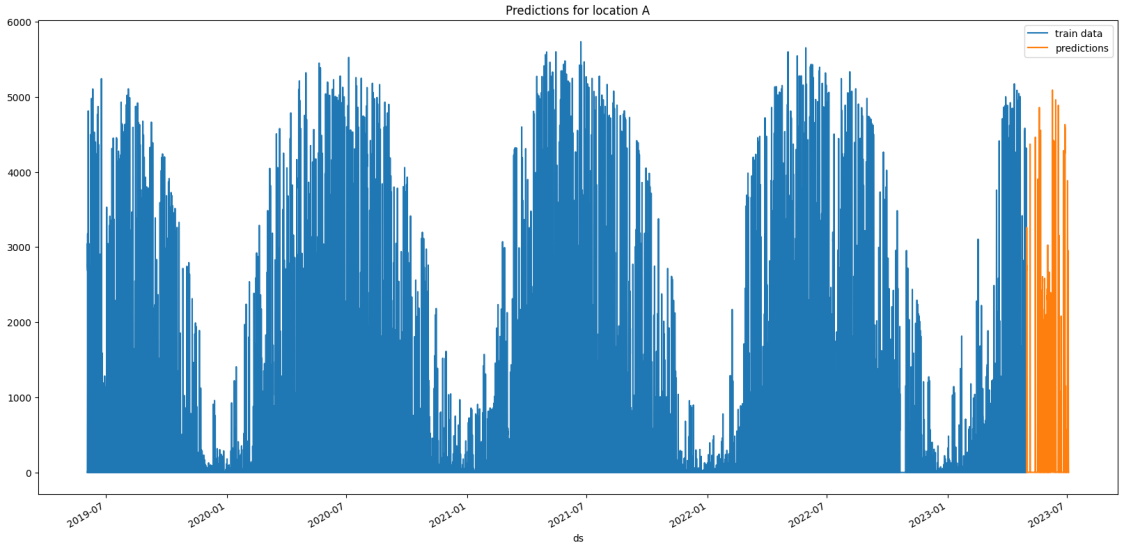
```
Loaded data from: test.csv | Columns = 4 / 4 | Rows = 2160 -> 2160
```

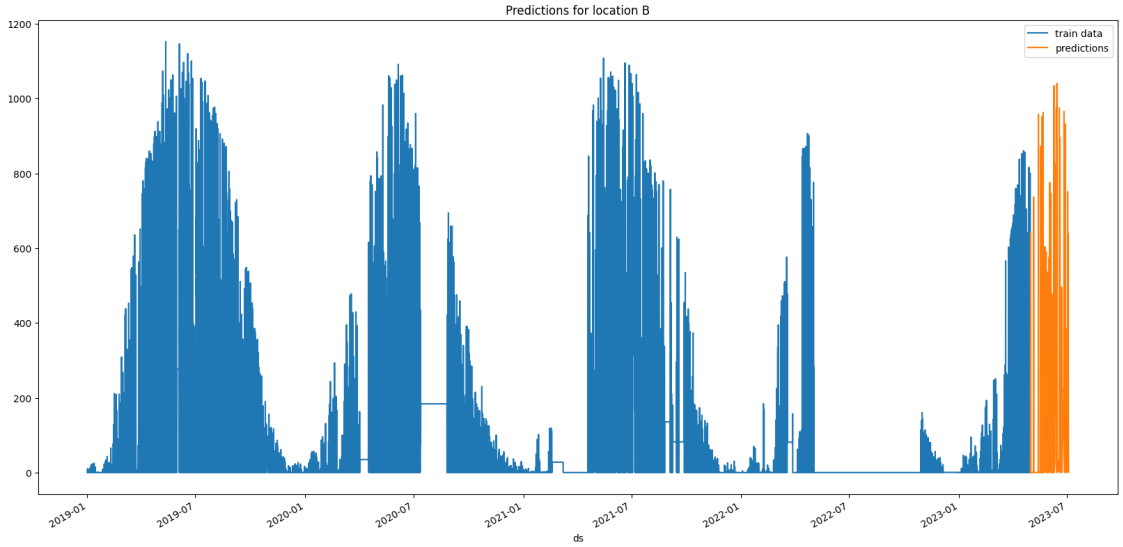
[13]:
```
# predict, grouped by location
predictions = []
location_map = {
    "A": 0,
    "B": 1,
    "C": 2
}
for loc, group in test_data.groupby('location'):
    i = location_map[loc]
    subset = test_data_merged[test_data_merged["location"] == loc].
 ↪reset_index(drop=True)
    #print(subset)
    pred = predictors[i].predict(subset)
    subset["prediction"] = pred
    predictions.append(subset)
```

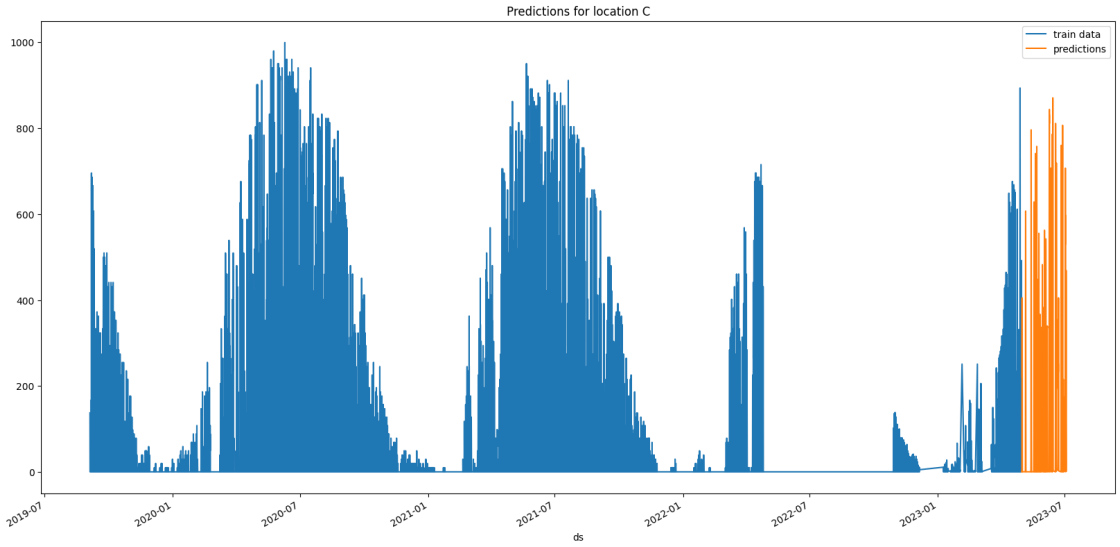
[14]:
```
# plot predictions for location A, in addition to train data for A
for loc, idx in location_map.items():
    fig, ax = plt.subplots(figsize=(20, 10))
    # plot train data
    train_data_with_dates[train_data_with_dates["location"]==loc].plot(x='ds',␣
 ↪y='y', ax=ax, label="train data")
```

```
    # plot predictions
    predictions[idx].plot(x='ds', y='prediction', ax=ax, label="predictions")

    # title
    ax.set_title(f"Predictions for location {loc}")
```







[15]:
```
# concatenate predictions
submissions_df = pd.concat(predictions)
submissions_df = submissions_df[["id", "prediction"]]
submissions_df
```

[15]:

	id	prediction
0	0	1.474567
1	1	1.537354
2	2	1.680374
3	3	47.797668
4	4	300.030823
..	…	…
715	2155	83.732719
716	2156	61.742329
717	2157	29.696980
718	2158	3.763743
719	2159	2.140930

[2160 rows x 2 columns]

[16]:
```
# Save the submission DataFrame to submissions folder, create new name based on␣
 ↪last submission, format is submission_<last_submission_number + 1>.csv

# Save the submission
print(f"Saving submission to submissions/{new_filename}.csv")
submissions_df.to_csv(os.path.join('submissions', f"{new_filename}.csv"),␣
 ↪index=False)
```

```
Saving submission to submissions/submission_81_jorge.csv
```

[17]:
```
# save this notebook to submissions folder
import subprocess
import os
subprocess.run(["jupyter", "nbconvert", "--to", "pdf", "--output", os.path.
 ↪join('notebook_pdfs', f"{new_filename}.pdf"), "autogluon_each_location.
 ↪ipynb"])
```

```
[NbConvertApp] Converting notebook autogluon_each_location.ipynb to pdf
[NbConvertApp] Support files will be in notebook_pdfs/submission_81_jorge_files/
[NbConvertApp] Making directory
./notebook_pdfs/submission_81_jorge_files/notebook_pdfs
[NbConvertApp] Writing 121410 bytes to notebook.tex
[NbConvertApp] Building PDF
[NbConvertApp] Running xelatex 3 times: ['xelatex', 'notebook.tex', '-quiet']
[NbConvertApp] Running bibtex 1 time: ['bibtex', 'notebook']
[NbConvertApp] WARNING | bibtex had problems, most likely because there were no
citations
[NbConvertApp] PDF successfully created
[NbConvertApp] Writing 372019 bytes to notebook_pdfs/submission_81_jorge.pdf
```

[17]:
```
CompletedProcess(args=['jupyter', 'nbconvert', '--to', 'pdf', '--output',
'notebook_pdfs/submission_81_jorge.pdf', 'autogluon_each_location.ipynb'],
returncode=0)
```

[23]:
```
# feature importance
location="A"
split_time = pd.Timestamp("2022-10-28 22:00:00")
estimated = train_data_with_dates[train_data_with_dates["ds"] >= split_time]
estimated = estimated[estimated["location"] == location]
predictors[0].feature_importance(feature_stage="original", data=estimated,␣
 ↪time_limit=60*10)
```

```
These features in provided data are not utilized by the predictor and will be
ignored: ['location']
Computing feature importance via permutation shuffling for 48 features using
1440 rows with 5 shuffle sets…
        639.68s = Expected runtime (127.94s per shuffle set)
```

[ ]:
```
# feature importance
observed = train_data_with_dates[train_data_with_dates["ds"] < split_time]
observed = observed[observed["location"] == location]
predictor.feature_importance(feature_stage="original", data=observed,␣
 ↪time_limit=60*10)
```

[6]:
```
subprocess.run(["jupyter", "nbconvert", "--to", "pdf", "--output", os.path.
 ↪join('notebook_pdfs', f"{new_filename}_with_feature_importance.pdf"),␣
 ↪"autogluon_each_location.ipynb"])
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
/Users/skog/Documents/1-2023-autumn/school/TDT4173-machine-learning/project/
↪TDT4173/autogluon_each_location.ipynb Cell 22 line 1
----> <a href='vscode-notebook-cell:/Users/skog/Documents/1-2023-autumn/school/
↪TDT4173-machine-learning/project/TDT4173/autogluon_each_location.
↪ipynb#X30sZmlsZQ%3D%3D?line=0'>1</a> subprocess.run(["jupyter", "nbconvert",␣
↪"--to", "pdf", "--output", os.path.join('notebook_pdfs',␣
↪f"{new_filename}_with_feature_importance.pdf"), "autogluon_each_location.
↪ipynb"])

NameError: name 'subprocess' is not defined
```

[ ]:
```python
import subprocess

def execute_git_command(directory, command):
    """Execute a Git command in the specified directory."""
    try:
        result = subprocess.check_output(['git', '-C', directory] + command,␣
 ↪stderr=subprocess.STDOUT)
        return result.decode('utf-8').strip(), True
    except subprocess.CalledProcessError as e:
        print(f"Git command failed with message: {e.output.decode('utf-8').
 ↪strip()}")
        return e.output.decode('utf-8').strip(), False

git_repo_path = "."

branch_name = new_filename

# add datetime to branch name
branch_name += f"_{pd.Timestamp.now().strftime('%Y-%m-%d_%H-%M-%S')}"

commit_msg = "run result"

execute_git_command(git_repo_path, ['checkout', '-b',branch_name])

# Navigate to your repo and commit changes
execute_git_command(git_repo_path, ['add', '.'])
execute_git_command(git_repo_path, ['commit', '-m',commit_msg])

# Push to remote
output, success = execute_git_command(git_repo_path, ['push',␣
 ↪'origin',branch_name])

# If the push fails, try setting an upstream branch and push again
```

```
if not success and 'upstream' in output:
    print("Attempting to set upstream and push again...")
    execute_git_command(git_repo_path, ['push', '--set-upstream',␣
 ↪'origin',branch_name])
    execute_git_command(git_repo_path, ['push', 'origin', 'henrik_branch'])
```